

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Serial No. .... 10/659,221  
Filing Date ..... 09/10/2003  
Confirmation No. .... 5751  
Inventorship ..... James R. Larus  
Applicant ..... Microsoft Corporation  
Group Art Unit..... 2191  
Examiner ..... Ryan Coyer  
Attorney's Docket No. .... MS1-4497US  
Title: ..... Contracts and futures in an asynchronous programming language

**INFORMAL CLAIM LISTING FOR EXAMINER'S AMENDMENT PER  
PHONE CONVERSATIONS**

To: Commissioner of Patents and Trademarks  
PO Box 1450  
Alexandria, VA 22313-1450

From: Beatrice L. Koempel-Thomas  
(Tel. 509-324-9256; Fax 509-323-8979)  
Lee & Hayes, PLLC  
601 W. Riverside Avenue, Suite 1400  
Spokane, WA 99201  
Customer Number 22801

**Brief Summary of Selected Substantive Portions of this Response**

**[0001]** In telephone discussions (on June 4, 2009 and June 9, 2009), the Examiner suggested an Examiner's Amendment to put claims into condition for allowability. A Word™ document listing of the proposed claims is included herein.

## AMENDMENTS

### **In the Claims**

**1. (Examiner Proposed Amended)** A system that facilitates asynchronous programming, comprising:

at least one processor that executes the following computer executable components stored on at least one computer readable medium:

a contract component that facilitates stating a behavioral contract defined according to an object oriented programming language on an asynchronous service interface between a client interface and a service interface; and

a conformance checking component that checks that at least one of the client interface and the service interface conform to the contract defined therebetween; and

a model extraction component that automatically extracts, on an individual function level, a client model and a service model from the respective interfaces such that a model checking component may systematically explore all of the possible interactions between the client and the service.

**2. (Original)** The system of claim 1, the contract defines an order in which clients of the service interface are invoked.

3. **(Original)** The system of claim 2, the checking component checks that the clients handle both normal and exceptional results.

4. **(Examiner Proposed Amended)** The system of claim 1, at least one of wherein the asynchronous service interface and the contract are defined according to an objected oriented program language.

5. **(Original)** The system of claim 1, the checking by the checking component is modular wherein the contract is specified on an interface boundary of the client interface and the service interface and, each of the client interface and the service interface is checked independently, and each method is checked separately.

6. **(Original)** The system of claim 1, the checking component further comprises a model extraction component that automatically extracts a behavior model that is fed to a model checker component.

7. **(Original)** The system of claim 6, the extracted model is sound in the presence of aliasing.

8. **(Original)** The system of claim 1, further comprising a construct in the form of an asynchronous call that returns a future-like handle to an eventual result of the call.

9. **(Original)** The system of claim 1, further comprising a construct in the form of a synch statement.

10. **(Original)** The system of claim 1, further comprising a construct that states the behavioral contract on the service interface.

11. **(Original)** The system of claim 1, further comprising a construct that is a transaction directive that delineates the scope of conversations by clients.

12. **(Original)** The system of claim 1, further comprising a construct that is a tracked type qualifier used by the checking component.

13. **(Original)** The system of claim 1, further comprising a procedure, which procedure is a method that is called at least one of synchronously and asynchronously.

**14. (Original)** The system of claim 13, the asynchronous call executes concurrently with its caller by completing immediately, while the method that is invoked executes.

**15. (Original)** The system of claim 14, the asynchronous call completes by returning a value.

**16. (Original)** The system of claim 15, the value is used explicitly to synchronize the asynchronous call with the value.

**17. (Original)** The system of claim 15, the value is a first-class type that can be at least one of stored, passed as an argument, and returned as a result.

**18. (Original)** The system of claim 17, the result returned is explicitly retrieved through a join statement.

**19. (Original)** The system of claim 15, the value is an object in one of three states that include an undefined state, normal state, and exceptional state.

**20. (Original)** The system of claim 1, the contract is a source-level contract that facilitates statically detecting asynchronous message passing errors.

21. **(Original)** The system of claim 1, the contract expresses stateful protocols that govern interactions between a client associated with the client interface and a service associated with the service interface.

22. **(Original)** The system of claim 1, the service interface is associated with a service contract whose language models the effects of multiple clients that concurrently access the service interface.

23. **(Original)** The system of claim 1, the checking component checks that services and clients obey the contract associated with the service interface.

24. **(Original)** The system of claim 1, the checking component uses a transaction directive to delimit a scope of an instance of a concurrent interaction between a client and a set of service instances.

25. **(Original)** The system of claim 24, the transaction directive specifies a correlation variable that uniquely identifies the instance of the concurrent interaction between the client and the set of service instances.

26. **(Original)** The system of claim 1, the checking component is modular in that to check for the client conformance and the service conformance, only the associated contract needs to be referenced.

27. **(Original)** The system of claim 1, the checking component considers a method as a function of an object to which it belongs.

28. **(Original)** The system of claim 1, the checking component relates a contract state, a client state, and a service state.

29. **(Original)** The system of claim 28, the checking component further comprises an extraction component that extracts a client model, service model, and interface model in order to relate the contract state, client state, and service state.

30. **(Original)** The system of claim 28, the checking component further comprises an extraction component that uses a region to model relevant data, which relevant data is that data which is directly relevant to the contract state.

31. **(Original)** The system of claim 30, the region is polymorphic.

32. **(Original)** The system of claim 30, the region is partial such that only relevant data is directly assigned to a region type.

33. **(Original)** The system of claim 1, the checking component further comprises an extraction component that facilitates model reduction by utilizing only necessary statements.

34. **(Original)** The system of claim 1, the checking component further comprises an extraction component that utilizes region-and-effect analysis to extract a model.

35. **(Original)** The system of claim 1, the checking component further comprises an extraction component that utilizes type-and-effect inference and source-level tracked types.

36. **(Original)** The system of claim 1, the contract is programmer-specified to separately check client and server code.

37. **(Original)** A computer according to the system of claim 1.

**38. (Examiner Proposed Amended)** A system that facilitates asynchronous programming, comprising:

at least one processor that executes the following computer executable components stored on at least one computer readable medium:

an extraction component that automatically extracts a client model from a client interface and a service model from a service interface on an individual function level, such that a model checking component may systematically explore all of the possible interactions between the client and the service;

a relator component that creates a behavioral relationship contract defined according to an object oriented programming language between the client model and the service model; and

a conformance checking component that checks that the client interface obeys the behavioral relationship contract, and the service interface properly implements the behavioral relationship contract.

**39. (Proposed Amended)** The system of claim 38, the behavioral relationship contract expressed in terms of at least one of a future, a join on the future, and asynchronous function calls.

40. **(Examiner Proposed Canceled)** The system of claim 38, the behavioral relationship is a contract the terms of which are enforced by the checking component.

41. **(Proposed Amended)** The system of claim 38, the conformance checking component checks that clients of the client interface are invoked in the proper order and that the clients handle both normal and exceptional results.

42. **(Original)** The system of claim 38, the extraction component creates the client model that defines relevant data for interaction with the service interface.

43. **(Original)** The system of claim 38, the extraction component creates the service model that defines relevant data for interaction with the client interface.

44. **(Proposed Amended)** The system of claim 38, the relator component checks the client model and the service model such that contract descriptions of the behavioral relationship contract are defined between the client model and the service model.

**45. (Examiner Proposed Amended)** A method of asynchronous programming, comprising:

employing a processor to execute computer readable instructions stored on a computer readable medium to perform the following acts:

receiving a client interface and a service interface;

creating a behavioral relationship contract defined according to an object oriented programming language between the client interface and the service interface; and

checking to ensure that the client interface obeys the behavioral relationship contract and the service interface properly implements the behavioral relationship contract; and

automatically extracting via a model extraction component, on an individual function level, a client model and a service model from the respective interfaces such that a model checking component may systematically explore all of the possible interactions between the client and the service.

**46. (Proposed Amended)** The method of claim 45, the behavioral relationship contract is expressed in terms of at least one of a future, a join on the future, and asynchronous function calls.

47. **(Original)** The method of claim 45, the behavioral relationship is a contract the terms of which are enforced by a modular checking component.

48. **(Original)** The method of claim 47, the modular checking component checks that clients of the client interface are invoked in the proper order and that the clients handle both normal and exceptional results.

49. **(Proposed Canceled)** ~~The method of claim 45, further comprising extracting with a model extraction component at least one of a client model for the client interface and a service model for the service interface.~~

50. **(Proposed Amended)** The method of claim [[49]] 45, the model extraction component creating the client model that defines relevant data for interaction with the service interface.

51. **(Proposed Amended)** The method of claim [[49]] 45, the model extraction component creates the service model that defines relevant data for interaction with the client interface.

**52. (Proposed Amended)** The method of claim [[49]] 45, further comprising performing model checking of the client model and the service model with a model checker such that contract descriptions are defined between the client model and the service model.

**53. (Proposed Amended)** The method of claim 45, further comprising automatically extracting the behavioral relationship contract.

**54. (Examiner Proposed Amended)** A method of asynchronous programming, comprising:

employing a processor to execute computer readable instructions stored on a computer readable medium to perform the following acts:

receiving a client interface and a service interface;

automatically extracting a client model from the client interface and a service model from the service interface on an individual function level, such that a model checking component may systematically explore all of the possible interactions between the client and the service;

creating a behavioral relationship contract defined according to an object oriented programming language between the client model and the service model; and

enforcing the behavioral relationship contract between the client interface and the service interface.

**55. (Proposed Amended)** The method of claim 54, the behavioral relationship is contract comprises a contract expressed by constructs in terms of at least one of a future, a join on the future, and asynchronous function calls, and the terms of which are enforced by a modular checking component.

**56. (Original)** The method of claim 54, further comprising checking that clients of the client interface are invoked in the proper order and that the clients handle both normal and exceptional results.

**57. (Proposed Amended)** The method of claim 54, wherein further comprising: extracting with a model extraction component a client model from the client interface, the client model defines relevant data for interaction with the service interface for the client interface,[[;]] and extracting with the model extraction component a service model from the service interface, the service model defines relevant data for interaction with the client interface.

**58. (Examiner Proposed Amended)** An article of manufacture tangibly embodying program instructions for execution by a processing unit, the processing unit retrieving the instructions from a computer-readable memory to facilitate concurrent execution of the instructions, the product comprising:

a contract component that creates a contract state defined according to an object oriented programming language between a client state and a service state;

an extraction component that automatically extracts a client model from the client state, a service model from the service state, and an interface model from the contract state on an individual function level, such that a conformance checking component may systematically explore all of the possible interactions between the client and the service;

and [[a]] the conformance checking component that checks that the client model obeys the interface model, and the service model properly implements the contract model.

**59. (Previously Presented)** The article of claim 58, the client model, service model, and interface model contain only respective communication actions and operations that are relevant for maintaining the state of a communication protocol.

**60. (Previously Presented)** The article of claim 59, the relevant operations are represented by the use of regions.

**61. (Previously Presented)** The article of claim 58, the interface model includes one or more effect processes for each method of the interface state.

**62. (Previously Presented)** The article of claim 61, the effect process models a method of the interface state according to an effect a call has on the contract state, a future created by an asynchronous call to the method, and futures passed into the method.

**63. (Examiner Proposed Amended)** A system of asynchronous programming, comprising one or more computing devices configured with:

means for receiving a client interface and a service interface;

means for automatically extracting a client model from the client interface and a service model from the service interface on an individual function level, such that a means for checking may systematically explore all of the possible interactions between the client and the service;

means for creating a behavioral relationship contract defined according to an object oriented programming language between the client model and the service model; and

the means for checking to ensure that the client interface obeys the behavioral relationship contract and the service interface properly implements the behavioral relationship contract.

**64. (Previously Presented)** The system of claim 18, the first class type is introduced by one of a local declaration or a field member declaration.

**REMARKS**

**[0002]** Applicant thanks the Examiner for his reconsideration and suggestion of an Examiner's Amendment for allowance of the pending claims.

**[0003]** Applicant understands that the Examiner has suggested amendments as presented above. Without conceding that such amendments are necessary, Applicant agrees to the Examiner's suggested amendments, and earnestly solicits allowance of the claims of this application. Amendments presented other than as suggested by the Examiner are merely to address minor antecedence or other grammatical inconsistencies consistent with the Examiner proposed amendments.

**Formal Request for an Interview**

**[0004]** If the Examiner's reply to this communication is anything other than allowance of all pending claims, then I formally request an interview with the Examiner. I encourage the Examiner to call me—the undersigned representative for the Applicant—so that we can talk about this matter so as to resolve any outstanding issues quickly and efficiently over the phone. Please contact me or my assistant to schedule a date and time for a telephone interview that is most convenient for both of us. While email works great for us, I welcome your call to either of us as well. Our contact information may be found below.

**Conclusion**

**[0005]** Applicant respectfully requests prompt issuance of the application. If any issues remain that prevent issuance of this application, the **Examiner is urged to contact me before issuing a subsequent Action**. Please call/email me or my assistant at your convenience.

Respectfully Submitted,

Dated: June 10, 2009 By:

/Bea Koempel-Thomas 58213/  
Beatrice L. Koempel-Thomas  
Reg. No. 58213  
(509) 944-4759  
bea@leehayes.com  
www.leehayes.com

Assistant: Cherri Simon  
(509) 944-4776  
cherri@leehayes.com